CHABADA-CN作品研究报告

CHABADA-CN *

July 6, 2018

1 研究背景

检查应用程序是否执行它所声称的行为是一个长期困扰着开发者的问题。不幸的是，目前它也成为了困扰计算机使用者的问题。每当我们安装一个新的应用的时候，我们也承担着这是一个恶意应用的风险。

目前检测恶意应用的研究都集中在静态代码检测和动态行为分析上面。2010年TaintDroid[3]提出了一种基于信息流的恶意软件检测方式，它使用系统级别的动态污点跟踪和分析技术来跟踪多个敏感数据源，并且利用Android的虚拟化执行环境提供实时分析。2013年，DroidAPIMiner[1]使用基于API的恶意软件检测的方式，分析一批良性和恶意软件样本，为每一个样本生成一个API列表，挑选出在恶意样本中使用较多的API的集合，根据这个API集合来过滤恶意软件。2014年，Apposcopy[4]提出基于签名和信息流的恶意软件检测方式，它结合了基于模式的恶意软件检测器和污点分析器的优势，使用“描述Android恶意软件系列语义特征”和“为给定应用程序匹配恶意软件家族签名”这两种方式来检测恶意软件行为。2014年Drebin[2]使用机器学习的方式对恶意软件进行检测，它使用广泛静态分析的方式，尽可能地收集应用程序的功能，将这些功能嵌入在一个联合向量空间中，使用这个向量空间来自动识别恶意软件的典型模式。2017年，MaMaDroid[5]使用了一种基于行为分析的恶意软件检测方式，它首先构建应用程序API的调用序列，然后用马尔科夫链的形式构建行为模型，并用它来提取特征并执行软件分类。但是以上这些分析方法并不能抵抗新的攻击方式，因此CHABADA提出了一种根据应用程序描述来检查应用行为的方式。如果应用程序的行为未在其描述中出现，就有可能是恶意行为。CHABADA根据应用程序的描述信息将应用程序聚类，属于同一类的应用程序如果出现了不同于其他应用程序的行为，那么这个行为就可能是恶意的。比如在“导航和旅行”群集中，使用任何API改变电话或其组件的配置将是不寻常的。

由于CHABADA是针对英文应用实现的一种方案，我们的工作是将论文中的方法复现，并调整参数使其适用于中文描述的应用。我们使用了一个包含30万应用描述信息的数据集进行实验。

2 描述预处理

在进行后续操作之前，我们首先使用自然语言处理（NLP）的标准技术对描述进行分词、过滤等操作，将描述信息用词向量表示。

2.1 非中文过滤

数据集中的应用描述可能会包含多语言的段落。例如：主要描述是中文，而在描述中穿插着一些英文、日文、韩文等来描述具体的应用信息，这种情况在一些游戏或语言教学应用中非常常见。为了能够对类似的描述进行聚类，我们必须选择一种语言，在这里我们选择了中文。为了删除所有不是中文的文本段落，我们使用正则表达式来过滤非中文字符。

```
filtrate = re.compile(u'[^\u4E00-\u9FA5, 。]')
```

Figure 1: 正则表达式

2.2 分词

英文是以词为单位的，词和词之间是靠空格隔开，而中文是以字为单位，句子中所有的字连起来才能描述一个意思。例如，英文句子I am a student，用中文则为：“我是一个学生”。计算机可以很简单通过空格知道student是一个单词，但是不能很容易明白“学”、“生”两个字合起来才表示一个

*作者介绍：贺弋玲、刘怡萍、季雨娇、王梦媛、赵彦杰

词。把中文的汉字序列切分成有意义的词，就是中文分词，有些人也称为切词。我是一个学生，分词的结果是：我/是/一个/学生。

中文分词(Chinese Word Segmentation) 指的是将一个汉字序列切分成一个一个单独的词。分词就是将连续的字序列按照一定的规范重新组合成词序列的过程。

我们选用了jieba分词工具。jieba支持三种分词模式：

- 精确模式，试图将句子最精确地切开，适合文本分析；
- 全模式，把句子中所有的可以成词的词语都扫描出来, 速度非常快，但是不能解决歧义；
- 搜索引擎模式，在精确模式的基础上，对长词再次切分，提高召回率，适合用于搜索引擎分词。

jieba采用的算法是：

- 基于前缀词典实现高效的词图扫描，生成句子中汉字所有可能成词情况所构成的有向无环图 (DAG)
- 采用了动态规划查找最大概率路径, 找出基于词频的最大切分组合
- 对于未登录词，采用了基于汉字成词能力的 HMM 模型，使用了 Viterbi 算法

2.3 去停用词

停用词是指在信息检索中，为节省存储空间和提高搜索效率，在处理自然语言数据（或文本）之前或之后会自动过滤掉某些字或词，这些字或词即被称为Stop Words（停用词）。这些停用词都是人工输入、非自动化生成的，生成后的停用词会形成一个停用词表。但是，并没有一个明确的停用词表能够适用于所有的工具。甚至有一些工具是明确地避免使用停用词来支持短语搜索的。

停用词分为以下两类：

- 使用十分广泛，甚至是过于频繁的一些单词。比如英文的“i”、“is”、“what”，中文的“我”、“就”、“的”、“个”之类词几乎在每个文档上均会出现，这样的词不具备代表一个文档的功能。在用向量表示文档时，如果我们使用了太多的停用词，可能无法得到非常好的效果，甚至是可能大量毫不相关的结果。
- 文本中出现频率很高，但实际意义又不大的词。这一类主要包括了语气助词、副词、介词、连词等，通常自身并无明确意义，只有将其放入一个完整的句子中才有一定作用的词语。如常见的“的”、“在”、“和”、“接着”之类，比如“吴宣仪是一个漂亮的小姑娘”这句话中的“是”、“的”就是两个停用词。

我们将对“哈工大停用词词库”、“四川大学机器学习智能实验室停用词库”、“百度停用词表”等等各种停用词表整理去重后得到一份停用词表，一共包含1598个停用词。以此对分词后的结果进行去停用词处理。

下图为NLP处理后的效果示例图

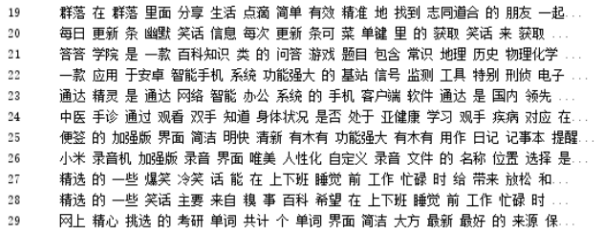

Figure 2: NLP效果示例图

3 使用LDA识别主题

3.1 简介

LDA（Latent Dirichlet Allocation）是由 Blei 等人于 2003 年提出的概率增长模型，是一种文档主题生成模型，也称为一个三层贝叶斯概率模型，包含词、主题和文档三层结构。

LDA是一种非监督机器学习技术，不需要训练大量数据，可以用来识别大规模文档集（document collection）或语料库（corpus）中潜藏的主题信息。相比于关键词匹配技术， LDA 主题模型的优点在于：LDA 模型更关注文档的语义信息，是一种抽象层次更高的匹配技术。模型的符号及其说明如表所列。

LDA是一种典型的词袋模型，即它认为一篇文档是由一组词构成的一个集合，词与词之间没有顺序以及先后的关系。一篇文档可以包含多个主题，文档中每一个词都由其中的一个主题生成。LDA的生成流程图如算法1所示。

因此整个模型中所有可见变量以及隐藏变量的

Algorithm 1 LDA生成流程

1.从狄利克雷分布α中取样生成文档i的主题分布θ_i

2.从主题的多项式分布 θ_i中取样生成文档i第j个词的主题 $z_{i,j}$

3.从狄利克雷分布 β中取样生成主题$z_{i,j}$的词语分布$\phi_{z_{i,j}}$

4.从词语的多项式分布$\phi_{z_{i,j}}$中采样最终生成词语$w_{i,j}$

联合分布是

$$p(w_i, z_i, \theta_i, \Phi|\alpha, \beta)$$
$$= \prod_{j=1}^{N} p(\theta_i|\alpha)p(z_{i,j}|\theta_i)p(\Phi|\beta)p(w_{i,j}|\phi_{z_{i,j}})$$

最终一篇文档的单词分布的最大似然估计可以通过将上式的θ_i以及Φ进行积分和对 z_i进行求和得到

$$p(w_i|\alpha, \beta) = \int_{\theta_i} \int_{\Phi} \sum_{z_i} p(w_i, z_i, \theta_i, \Phi|\alpha, \beta)$$

根据 $p(w_i|\alpha, \beta)$的最大似然估计，最终可以通过EM算法等方法估计出模型中的参数。

3.2 LDA在本次模型中的应用

我们将经过自然语言处理后的中文版应用描述词语转换成词频矩阵，例如矩阵中包含一个元素$a_{i,j}$，它表示j词在i类文本下的词频。将词频矩阵作为LDA模型的输入[8]。我们预先选择了30个主题，所以LDA模型将会输出主题及其对应词参数矩阵（主题-词参数），文本分别属于30个主题的概率矩阵（文本-主题），例如文本-主题矩阵中包含一个元素$b_{i,j}$，它表示文本i属于主题j的概率。其中LDA输出的文本-主题概率矩阵将会作为文本K-Means聚类的输入。表1为使用LDA模型生成的30个主题。

Table 1: LDA主题

ID	代表性词语
0	中国、平台、美食、商城、在线、文化、包括
1	经典、设计、完美、资源、特色、品牌、城市
2	保存、攻略、酒店、选择、公主、美容、提供
3	游戏、玩家、玩法、模式、画面、挑战、简单
4	孩子、学习、音乐、播放、动物、电脑、数字
5	手游、角色、敌人、武器、超级、体验、全新
6	世界、宝宝、有趣、测试、快乐、生活、成长
7	特效、卸载、手机、提示、桌面主题、流畅、界面
8	应用程序、选择、朋友、地图、享受、时间、位置
9	效果、声音、不用、可爱、描述、美丽、自动
10	时尚、购买、潮流、电子商务、小编、应用
11	儿童、故事、提醒、调整、开发、适合、患者
12	专业、学习、展示、介绍、教育、知识、需求
13	免费、设备、创建、图像、商机、绿豆、共享
14	商品、商家、方便快捷、操作、简洁、界面
15	手机、视频、设置、自动、短信、显示、快速
16	精彩、用户、团队、推荐、打造、直播、关注
17	平台、企业、管理、管家、国内、第一、经营
18	照片、锁屏、图片、精美、制作、分享、解锁
19	服务、客户、互联网、活动、提供、相关、考试
20	实时、打造、系统、汽车、体验、在线、市场
21	旅游、时间、语言、旅行、医生、掌上、国家
22	屏幕、更新、设置、版本、交流、搜索、系统
23	生活、分享、阅读、朋友、社交、交易、投资
24	主题、海量、手机、选择、宝软、高清大图、匣子
25	桌面、魔秀、个性、时尚、免费、适合、插件
26	健康、查询、记录、助手、运动、计算、食谱
27	壁纸、动态、手机、发布、作品、最新、获取
28	信息、用户、客户端、手机、一键、最新、注册
29	提供、产品、服务、信息、行业、资讯、专业

4 使用K-means进行聚类

主题建模以一定的概率为每个主题分配一个应用程序描述。换句话说，每个应用程序的特征是每个主题的亲和度值（概率）向量。然而，我们想要的是得到具有相似描述的应用程序组，我们使用K-means这种最常见的聚类算法对app描述信息进行聚类。

K-means算法是最为经典的基于划分的聚类方法，是十大经典数据挖掘算法之一。K-means算法的基本思想是：以空间中k个点为形心进行聚类，对最靠近他们的对象归类。通过迭代的方法，逐次更新各簇的形心的值，直至得到最好的聚类结果。（形心可以是实际的点、或者是虚拟点）[6]。事先确定常数k，常数k意味着最终的聚类类别数。将事先输入的n个数据对象划分为k个聚类以便使得所获得的聚类满足：同一聚类中的对象相似度较高；而不同聚类中的对象相似度较小。聚类相似度是利用各聚类中对象的均值所获得一个“中心对象”（引力中心）来进行计算的。

该算法的最大优势在于简洁和快速。算法的关键在于初始中心的选择和距离公式。K-means的算法流程如算法2所示:

Algorithm 2 Kmeans算法流程

1.适当选择k个簇的初始形心

2.在第m次迭代中，对任意一个样本，求其到k个形心的欧氏距离或曼哈顿距离，将该样本归类到距离最小的形心所在的簇；

3.利用均值等方法更新该簇的形心值

4.对于所有的k个簇形心，如果利用2、3的迭代法更新后，当形心更新稳定或误差平方和最小时，则迭代结束，否则继续迭代。（误差平方和即簇内所有点到形心的距离之和）

图3为聚类结果示例图。

	A	B
1	包名	类别
2	ABCD.Words	31
3	ACGame.Yxz	4
4	ACloud.BPM	10
5	ACloud.BQYGL	24
6	ACloud.MindMap	2
7	ACloud.MindNote	16
8	ACloud.Trade	10
9	APPeal.NumberScrambler	25
10	AP_Images06.T	0
11	AT_Novel69.T	9
12	AdMaiora.SmashingPlanets	7
13	Adrenaline.Crew	3
14	Adrenaline.Crewrmb	25
15	Ai.Shinozaki.Mov	6
16	AlexBrowserPro.namespace	13
17	AndPoem.AndPoem	23
18	Andrew.Compass.com	8

Figure 3: 部分app的聚类结果

图4为Kmeans聚类得到的某个类别的部分应用展示。

在本次实验中，我们取k为32。

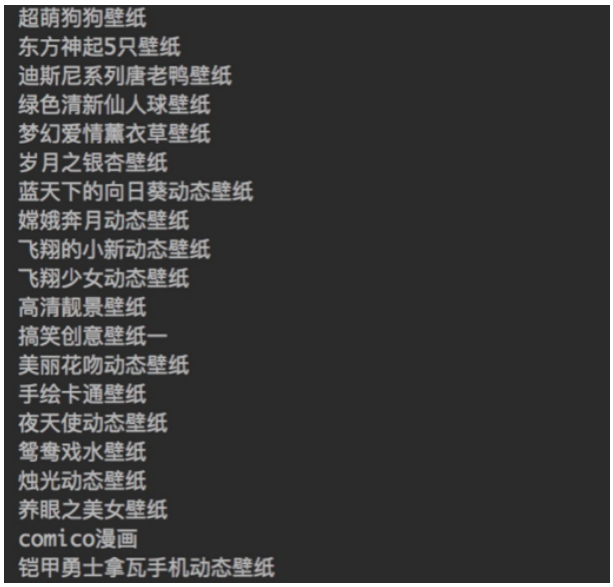

Figure 4: 某类别部分应用展示

5 使用OCSVM识别异常APP

5.1 OCSVM算法研究

ocsvm(one class SVM)，与传统SVM不同的是，one class SVM是一种非监督的算法。它是指在训练集中只有一类positive（或者negative）的数据，而没有另外的一类。而这时，需要学习的就是边界（boundary），而不是最大间隔（maximum margin）.用某一类样本（正样本或负样本）进行训练以后，ocsvm可以根据训练得到的边界将数据分为两类，所以ocsvm可以用来做异常检测。

标准的SVM算法的思路[7]是构造一个广义最优分类面，尽量使训练数据集中分属两类的数据点露在这个分类面的两边，并且两类的分类间隔尽可能大. 同样地，OCSVM算法假设坐标原点为异常的样本，在特征空间中构建一个最优的超平面实现目标数据与坐标原点的最大间隔. 目前的OCSVM算法主要分为两种：超球法和超平面法. 由Tax提出的超球法旨在找到一个包含所有数据并且体积最小的超球面；另一种方法是由Scholkopf等于1999年提出的超平面法，其主要思想是在特征空间中找到一个最优的超平面使数据对象与原点以最大距离分开. 实际上，当使用高斯核函数时，即：

$$K\left(x_i, x_j\right)=\left\langle\phi\left(x_i\right), \phi\left(x_j\right)\right\rangle=\exp\left(-g\left\|x_i-x_j\right\|^2\right)$$

这两种方法是相同的，其基本思想都是将输入空间通过核函数映射到高维空间，在高维空间中将它们与原点尽可能地分开。进而把分类问题转化成

一个特殊的二值分类问题。使得在囊括了大多数正常样本点的高密度区域中的样本点类别记作“+1”，而位于这个高密度区域之外的异常样本点类别记作“-1”。

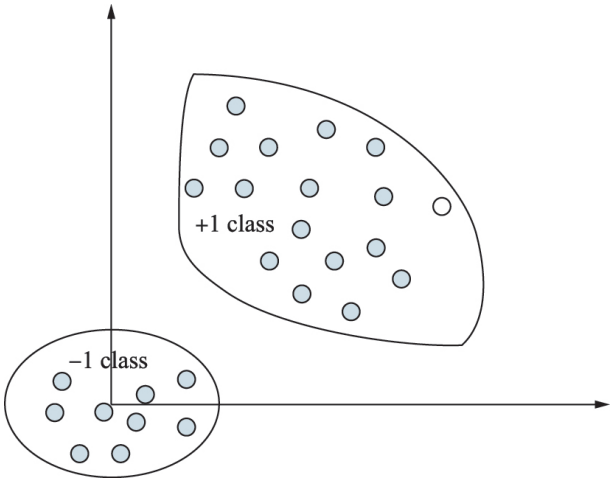

Figure 5: 单类支持向量机分类问题

OCSVM解决如下的二次规划问题:

$$\min_{\zeta \in R^i, p \in R} \frac{1}{2}\|\omega\|^2 + \frac{1}{vn}\sum_{i=1}^{n}\zeta_i - \rho$$

$$\phi(x_i)\,\omega \geq \rho - \zeta_i, \zeta_i \geq 0$$

其中，x_i为原空间中的样本，n为训练样本的数量，ϕ为原始空间到特征空间的映射，ω和ρ分别为特征空间中所需超平面的法向量和补偿，可调参数$v \in (0,1)$为控制总样本数中错误样本比例的上界，松弛变量ζ_i为某些训练样本被错误分类的程度. 如果ω和ρ是上面这个二次规划问题的解，则其决策函数为 :

$$f(x) = \operatorname{sgn}(\phi(x)\,\omega - \rho)$$

5.2 OCSVM在本模型中的应用

在chabada中，我们使用ocsvm来识别相应主题群中API使用情况异常的应用程序。我们搜集了所有APP使用的敏感API列表，根据K-means的聚类结果将同一类APP的敏感API使用情况生成一个矩阵，我们选取了此类中的一部分非恶意软件作为正样本对OCSVM进行训练，再把整个矩阵作为OCSVM的输入样本。OCSVM会将样本分为正样本和负样本两类，其中正样本表示同一类中大多数APP的敏感API使用情况，而负样本则表示该APP调用了与其描述行为不符的敏感API。图7是OCSVM的流程示意图。

Figure 6: OCSVM

图8中显示的是在某一类中使用OCSVM后的部分分类结果，从图中可以看出，这几个APP的描述基本一致，但是其中出现了一个样本被标记为负样本。

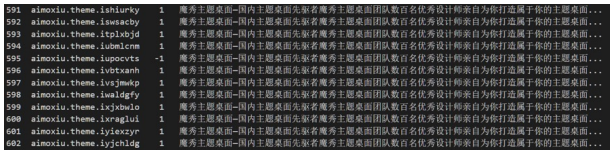

Figure 7: 异常检测结果

检查了图中所有样本的敏感API使用情况，发现只有被标记为负样本的APP所申请的敏感API远远多于该类别中其它APP所申请的敏感API，具体的敏感API调用情况如图9所示。

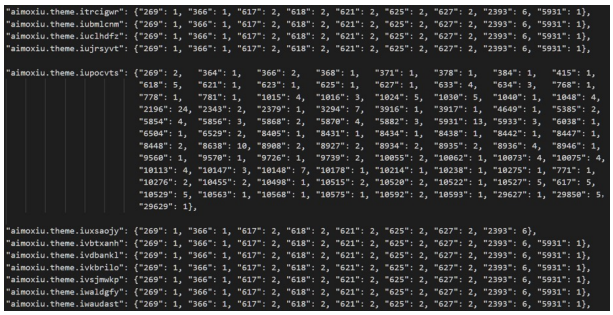

Figure 8: 敏感API示意图

为了将同一类别中的应用程序通过敏感API更准确地分为正负两种样本，即将其中的异常值检测出来，我们在调用sk-learn相关函数时，经过了长时间的参数调整。我们在数据集上尝试了高斯核和线性核两种核函数，以及0.1，0.2，0.3，0.4，0.5等不同nu值。最终我们选择使用高斯核以及nu=0.1来进行OCSVM分类器的训练。

6 结果评估

经过自然语言处理，LDA主题提取，K-means聚类，OCSVM分类，我们将所有的APP样本

Table 2: 第六类数据集检测结果

	预测为恶意	预测为良性
恶意应用程序	100%	0%
良性应用程序	20.3%	79.7%

Table 3: 全体数据集检测结果

	预测为恶意	预测为良性
恶意应用程序	33.7%	66.3%
良性应用程序	7.7%	92.3%

分为了正负两类样本，将标记为-1的样本视为可能的恶意软件，并将其与已判定为为恶意软件的数据集进行比较，得到使用chabada能够判断出的恶意软件的数量。以不同参数运行chabada得到的每类检测结果不尽相同，其中在ocsvm参数nu为0.5、核函数为rbf时，第六类结果最优。表2显示了此时第六类APP的恶意软件判定的结果。我们使用chabada准确地判断出了其中含有的151个恶意软件。

经过多次调参与大量程序化比较，我们最终确定的通用ocsvm参数是nu为0.1、核函数为rbf。表3显示了在我们总数为30万的APP，32个类别中恶意软件判定的平均结果。使用chabada能检测出的恶意软件

在理想情况下，表3中主对角线上的值，即恶意应用程序预测为恶意、良性应用程序预测为良性的百分比应该较高，第六类的结果就比较接近这一情况。在我们的CHABADA-CN中，良性应用的判定比较准确，而恶意应用的判定则有所欠缺。经过分析，我们列出了在没有先验知识的情况下，CHABADA-CN在拥有30万APP的数据集上运行结果出现误判的可能原因：

- 在一个比较大的malware dataset上面，使用以应用程序描述为特征聚类以及以敏感API为特征进行分类等方法进行异常检测也许本身并不是非常有效。
- 作为比较的恶意软件数据集，是virustotal的引擎结果中可能性较高的，只能作为参考。
- 我们工作的重点是在没有先验知识的情况下，尽可能的拎出异常使用API的APP，而这个异常结果并不一定是恶意的，恶意的应用也不一定存在于恶意结果中。

7 总结

通过主题描述对应用进行聚类，并根据每个集群内的API使用情况识别出异常值，我们的CHABADA方法可以有效地识别与其描述行为不符的应用程序。在我们的工作中，我们对Android应用程序生态系统有了很多认识。首先，应用程序供应商必须清楚地了解他们应用程序的具体功能。谷歌等应用商店供应商应该引入更好的标准，以避免欺骗或不完整的广告。其次，Android向用户请求权限的方式并不完整。普通用户不明白什么“允许访问设备标识符”的含义是什么，没有办法检查他们的敏感数据实际正在做什么，也不知道其后果。不过，如果用户了解常规应用的功能， CHABADA将指出与常规应用的突出差异，这应该更容易掌握。

References

[1] Yousra Aafer, Wenliang Du, and Heng Yin. *DroidAPIMiner: Mining API-Level Features for Robust Malware Detection in Android*. Springer International Publishing, 2013.

[2] Daniel Arp, Michael Spreitzenbarth, Malte Hübner, Hugo Gascon, and Konrad Rieck. Drebin: Effective and explainable detection of android malware in your pocket. In *Network and Distributed System Security Symposium*, 2014.

[3] William Enck, Peter Gilbert, Byung Gon Chun, Landon P. Cox, Jaeyeon Jung, Patrick Mcdaniel, and Anmol N. Sheth. Taintdroid: an information flow tracking system for real-time privacy monitoring on smartphones. *Acm Transactions on Computer Systems*, 32(2):1–29, 2014.

[4] Yu Feng, Saswat Anand, Isil Dillig, and Alex Aiken. Apposcopy: semantics-based detection of android malware through static analysis. In *ACM Sigsoft International Symposium on Foundations of Software Engineering*, pages 576–587, 2014.

[5] E. Mariconti, L. Onwuzurike, P. Andriotis, E De Cristofaro, G. Ross, and G. Stringhini. Mamadroid: Detecting android malware by building markov chains of behavioral models. In *NDSS '17: Net-*

work and Distributed Systems Security Symposium, 2017.

[6] 侯敬儒, 吴晟, and 李英娜. 基于spark streaming的在线kmeans聚类模型研究. 计算机与数字工程, (4), 2018.

[7] 尚文利, 李琳, 万明, and 曾鹏. 基于优化单类支持向量机的工业控制系统入侵检测算法. 信息与控制, 44(6):678–684, 2015.

[8] 李昌亚 and 刘方方. 基于lda的社科文献主题建模方法. 计算机技术与发展, (2):182–187, 2018.